package com.x00118478.www.weathercax00118478;  
  
import android.annotation.SuppressLint;  
import android.content.DialogInterface;  
import android.graphics.Typeface;  
import android.os.AsyncTask;  
import android.os.Bundle;  
import android.support.v7.app.AlertDialog;  
import android.support.v7.app.AppCompatActivity;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.LinearLayout;  
import android.widget.ProgressBar;  
import android.widget.TextView;  
import android.widget.Toast;  
import org.json.JSONException;  
import org.json.JSONObject;  
import java.text.DateFormat;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
import java.util.Locale;  
  
public class MainActivity extends AppCompatActivity {  
  
 TextView city;  
 TextView cityWeatherDescription;  
 Button changeCity;  
 TextView weather\_icon;  
 TextView weather\_conditions;  
 TextView wind\_direction;  
 TextView sunset;  
 TextView wind\_speed;  
 TextView maximum\_temperature;  
 TextView minimum\_temperature;  
 ProgressBar loading;  
 //Initialise the Weather graphics obtained from = http://erikflowers.github.io/weather-icons/  
 Typeface weatherGraphics;  
 String selectedCity = "Dublin, IE ";  
 //*TODO Generate API KEY and PASTE in the blank spot.* //Generated from an online weather API.  
 String API\_KEY = "649410531b3484189953fe198d9297fa";  
  
 @SuppressLint({"DefaultLocale", "SetTextI18n"})  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 //Fullscreen capability  
 getSupportActionBar().hide();  
 setContentView(R.layout.*activity\_main*);  
  
 //Connect the initialised variables to the interface by using the ID  
 city = findViewById(R.id.*city*);  
 cityWeatherDescription = findViewById(R.id.*updated*);  
 changeCity = findViewById(R.id.*changeCity*);  
 weather\_icon = findViewById(R.id.*weather\_icon*);  
 wind\_direction = findViewById(R.id.*wind\_direction*);  
 maximum\_temperature = findViewById(R.id.*maximum\_temperature*);  
 minimum\_temperature = findViewById(R.id.*minimum\_temperature*);  
 wind\_speed = findViewById(R.id.*wind\_speed*);  
 weather\_conditions = findViewById(R.id.*weather\_conditions*);  
 sunset = findViewById(R.id.*sunset*);  
 loading = findViewById(R.id.*loading*);  
 //Weather ICONS & Graphics are sourced from = http://erikflowers.github.io/weather-icons/ \*\*NOT MY PROPERTY  
 weatherGraphics = Typeface.*createFromAsset*(getAssets(), "icons/weathericons-regular-webfont.ttf");  
  
 weather\_icon.setTypeface(weatherGraphics);  
  
 downloadCityWeather(selectedCity);  
  
 city.setText((selectedCity).toUpperCase(Locale.*US*));  
 cityWeatherDescription.setText(("description").toUpperCase(Locale.*US*));  
 minimum\_temperature.setText("Min\n" + "8" + "°");  
 maximum\_temperature.setText("Max\n" + "20" + "°");  
 wind\_direction.setText("Wind Direction: " + "East");  
 wind\_speed.setText("Wind Speed: " + "5 " + "kmp/h");  
 weather\_conditions.setText("High");  
 sunset.setText("Low");  
  
 //Display Dialog box to allow the user to change the city.  
 //Dialog box based off https://developer.android.com/guide/topics/ui/dialogs  
 changeCity.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 AlertDialog.Builder dialogBox = new AlertDialog.Builder(MainActivity.this);  
 dialogBox.setTitle("Enter City Name:");  
 //declare EditText for the UI  
 final EditText userInputforCity = new EditText(MainActivity.this);  
 LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(  
 LinearLayout.LayoutParams.*MATCH\_PARENT*,  
 LinearLayout.LayoutParams.*MATCH\_PARENT*);  
 dialogBox.setView(userInputforCity);  
  
 //Confirm the change and retrieve the data from the API & Display  
 dialogBox.setPositiveButton("Change",  
 new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int which) {  
 selectedCity = userInputforCity.getText().toString();  
 userInputforCity.setText(selectedCity);  
 city.setText((selectedCity).toUpperCase(Locale.*US*));  
 downloadCityWeather(selectedCity);  
 }  
 });  
 //Cancel the requested change  
 dialogBox.setNegativeButton("Cancel",  
 new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int which) {  
 dialog.cancel();  
 }  
 });  
 dialogBox.show();  
 }  
 });  
  
 }  
  
 //Download the Weather information  
 public void downloadCityWeather(String query) {  
 if (Controller.*isDeviceConnected*(getApplicationContext())) {  
 CallAPI retrieveWeatherTask = new CallAPI();  
 retrieveWeatherTask.execute(query);  
 Toast.*makeText*(getApplicationContext(), "Great Success!", Toast.*LENGTH\_LONG*).show();  
 } else {  
 Toast.*makeText*(getApplicationContext(), "No Connection To The Internet.", Toast.*LENGTH\_LONG*).show();  
 }  
 }  
  
 //This is the backbone of the application  
 //Based off https://www.androidauthority.com/use-remote-web-api-within-android-app-617869/  
 class CallAPI extends AsyncTask<String, Void, String> {  
 @Override  
 //Retrieve the XML data from the API using the URL and the API Key.  
 protected String doInBackground(String... strings) {  
 String xml = Controller.*retrieveData*("http://api.openweathermap.org/data/2.5/weather?q=" + strings[0] +  
 "&units=metric&appid=" + API\_KEY);  
 return xml;  
 }  
  
 @Override  
 protected void onPreExecute() {  
 super.onPreExecute();  
 loading.setVisibility(View.*VISIBLE*);  
 }  
  
 @Override  
 protected void onPostExecute(String xml) {  
  
 try {  
 JSONObject json = new JSONObject(xml);  
 if (json != null) {  
 JSONObject weatherInformation = json.getJSONArray("weather").getJSONObject(0);  
 JSONObject mainIformation = json.getJSONObject("main");  
 JSONObject system = json.getJSONObject("sys");  
 JSONObject wind = json.getJSONObject("wind");  
 double windDegrees = wind.getDouble("deg");  
 //save the unix time stamp  
 long sunriseCode = system.getLong("sunrise");  
 long sunsetCode = system.getLong("sunset");  
 //convert to java timestamp  
 long javaSunrise = sunriseCode \* 1000L;  
 long javaSunset = sunsetCode \* 1000L;  
 Date date = new Date(javaSunrise);  
 Date dateSunset = new Date(javaSunset);  
 //Format to display the time in Hh:MM in twenty four hour format  
 String sunrise = new SimpleDateFormat("kk:mm").format(date);  
 String sunsetTime = new SimpleDateFormat("kk:mm").format(dateSunset);  
  
 city.setText(json.getString("name").toUpperCase(Locale.*US*) + ", " + json.getJSONObject("sys").getString("country"));  
 cityWeatherDescription.setText(weatherInformation.getString("description").toUpperCase(Locale.*US*));  
 minimum\_temperature.setText(String.*format*(" Min\n" + "%.2f", mainIformation.getDouble("temp\_min")) + "°");  
 maximum\_temperature.setText(String.*format*(" Max\n" + "%.2f", mainIformation.getDouble("temp\_max")) + "°");  
  
 if ((windDegrees >= 348.75) && (windDegrees <= 360) ||  
 (windDegrees >= 0) && (windDegrees <= 11.25)) {  
 wind\_direction.setText("Wind Direction: " + "North " + windDegrees + "°");  
 } else if ((windDegrees >= 11.25) && (windDegrees <= 33.75)) {  
 wind\_direction.setText("Wind Direction: " + "North-East " + windDegrees + "°");  
 } else if ((windDegrees >= 33.75) && (windDegrees <= 56.25)) {  
 wind\_direction.setText("Wind Direction: " + "North-East " + windDegrees + "°");  
 } else if ((windDegrees >= 56.25) && (windDegrees <= 78.75)) {  
 wind\_direction.setText("Wind Direction: " + "North-East " + windDegrees + "°");  
 } else if ((windDegrees >= 78.75) && (windDegrees <= 101.25)) {  
 wind\_direction.setText("Wind Direction: " + "East " + windDegrees + "°");  
 } else if ((windDegrees >= 101.25) && (windDegrees <= 123.75)) {  
 wind\_direction.setText("Wind Direction: " + "South-East " + windDegrees + "°");  
 } else if ((windDegrees >= 123.75) && (windDegrees <= 146.25)) {  
 wind\_direction.setText("Wind Direction: " + "South-East " + windDegrees + "°");  
 } else if ((windDegrees >= 146.25) && (windDegrees <= 168.75)) {  
 wind\_direction.setText("Wind Direction: " + "South-East " + windDegrees + "°");  
 } else if ((windDegrees >= 168.75) && (windDegrees <= 191.25)) {  
 wind\_direction.setText("Wind Direction: " + "South " + +windDegrees + "°");  
 } else if ((windDegrees >= 191.25) && (windDegrees <= 213.75)) {  
 wind\_direction.setText("Wind Direction: " + "South-West " + windDegrees + "°");  
 } else if ((windDegrees >= 213.75) && (windDegrees <= 236.25)) {  
 wind\_direction.setText("Wind Direction: " + "South-West " + windDegrees + "°");  
 } else if ((windDegrees >= 236.25) && (windDegrees <= 258.75)) {  
 wind\_direction.setText("Wind Direction: " + "South-West " + windDegrees + "°");  
 } else if ((windDegrees >= 258.75) && (windDegrees <= 281.25)) {  
 wind\_direction.setText("Wind Direction: " + "West " + windDegrees + "°");  
 } else if ((windDegrees >= 281.25) && (windDegrees <= 303.75)) {  
 wind\_direction.setText("Wind Direction: " + "North-West " + windDegrees + "°");  
 } else if ((windDegrees >= 303.75) && (windDegrees <= 326.25)) {  
 wind\_direction.setText("Wind Direction: " + "North-West " + windDegrees + "°");  
 } else if ((windDegrees >= 326.25) && (windDegrees <= 348.75)) {  
 wind\_direction.setText("Wind Direction: " + "North-West " + windDegrees + "°");  
 } else {  
 wind\_direction.setText("Wind Direction: " + " Not recognised ");  
 }  
  
 wind\_speed.setText("Wind Speed: " + wind.getString("speed") + "kmp/h");  
 weather\_conditions.setText("Sunrise :" + sunrise);  
 sunset.setText("Sunset : " + sunsetTime);  
 loading.setVisibility(View.*GONE*);  
  
 }  
 } catch (JSONException e) {  
 Toast.*makeText*(getApplicationContext(), "Error, City Not Found ", Toast.*LENGTH\_SHORT*).show();  
 }  
  
 }  
  
  
 }  
}

package com.x00118478.www.weathercax00118478;  
  
import android.content.Context;  
import android.net.ConnectivityManager;  
import android.net.NetworkInfo;  
  
import java.io.BufferedReader;  
import java.io.InputStream;  
import java.io.InputStreamReader;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
public class Controller {  
  
 //Retrieve the Network information from the device.  
 public static NetworkInfo retrieveNetworkInfo(Context context) {  
 ConnectivityManager connectionManager = (ConnectivityManager) context.getSystemService(Context.*CONNECTIVITY\_SERVICE*);  
 return connectionManager.getActiveNetworkInfo();  
 }  
  
 //Check if their is any internet connection.  
 public static boolean isDeviceConnected(Context context) {  
 NetworkInfo networkinfo = Controller.*retrieveNetworkInfo*(context);  
 return (networkinfo != null && networkinfo.isConnected());  
 }  
  
 public static String retrieveData(String urlTarget) {  
 URL url;  
 HttpURLConnection urlConnection = null;  
 try {  
 //Create connection  
 url = new URL(urlTarget);  
 urlConnection = (HttpURLConnection) url.openConnection();  
 urlConnection.setRequestProperty("content-type", "application/json; charset=utf-8");  
 urlConnection.setUseCaches(false);  
 urlConnection.setDoInput(true);  
 urlConnection.setDoOutput(false);  
  
 InputStream inputStream;  
 int statusCode = urlConnection.getResponseCode();  
 if (statusCode != HttpURLConnection.*HTTP\_OK*) {  
 inputStream = urlConnection.getErrorStream();  
 } else {  
 inputStream = urlConnection.getInputStream();  
 }  
 BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));  
 String line;  
 StringBuffer responseLine = new StringBuffer();  
 while ((line = bufferedReader.readLine()) != null) {  
 responseLine.append(line);  
 responseLine.append('\r');  
 }  
 bufferedReader.close();  
 return responseLine.toString();  
 } catch (Exception e) {  
 return null;  
 } finally {  
 if (urlConnection != null) {  
 urlConnection.disconnect();  
 }  
 }  
 }  
}

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/gradient"  
 android:padding="20dp">  
  
 <TextView  
 android:id="@+id/city"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceLarge" />  
  
 <TextView  
 android:id="@+id/updated"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/city"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceMedium"  
 android:textSize="13sp" />  
  
 <Button  
 android:id="@+id/changeCity"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:width="5dip"  
 android:text="@string/change\_city"  
 android:textStyle="bold|italic"  
 android:textSize="16sp"  
 android:textColor="@color/colorAccent"  
 android:background="@drawable/rectangle\_button"  
 android:layout\_below="@+id/updated"  
 android:layout\_centerHorizontal="true"/>  
  
  
 <TextView  
 android:id="@+id/weather\_icon"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerVertical="true"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:textSize="70sp"  
 />  
  
 <TextView  
 android:id="@+id/weather\_conditions"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/weather\_icon"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceMedium"  
 />  
 <TextView  
 android:id="@+id/sunset"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/weather\_conditions"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceMedium"  
 />  
 <TextView  
 android:id="@+id/wind\_direction"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/sunset"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceMedium"  
 />  
 <TextView  
 android:id="@+id/wind\_speed"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/wind\_direction"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceMedium"  
 />  
  
 <TextView  
 android:id="@+id/maximum\_temperature"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:layout\_alignParentRight="true"  
 android:textSize="40sp" />  
 <TextView  
 android:id="@+id/minimum\_temperature"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:textColor="@color/white"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:layout\_alignParentLeft="true"  
 android:textSize="40sp" />  
  
  
 <ProgressBar  
 android:id="@+id/loading"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerInParent="true"/>  
  
  
  
</RelativeLayout>

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.x00118478.www.weathercax00118478">  
 <uses-permission android:name="android.permission.INTERNET"/>  
 <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  
  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme">  
 <activity android:name=".MainActivity">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

<?xml version="1.0" encoding="utf-8"?>  
<selector xmlns:android="http://schemas.android.com/apk/res/android">  
  
 <item android:state\_pressed="true" >  
 <shape android:shape="rectangle" >  
 <corners android:radius="3dip" />  
 <stroke android:width="1dip" android:color="#77ccf9" />  
 <gradient android:angle="-90" android:startColor="#345953" android:endColor="#689a92" />  
 </shape>  
 </item>  
  
 <item android:state\_focused="true">  
 <shape android:shape="rectangle" >  
 <corners android:radius="3dip" />  
 <stroke android:width="1dip" android:color="#77ccf9" />  
 <solid android:color="#77ccf9"/>  
 </shape>  
 </item>  
  
 <item >  
 <shape android:shape="rectangle" >  
 <corners android:radius="3dip" />  
 <stroke android:width="2dip" android:color="#FFFFFF" />  
 <gradient android:angle="-90" android:startColor="@color/colorPrimary" android:endColor="@color/background" />  
 </shape>  
 </item>  
</selector>

API

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using Weather.Models;

namespace Weather.Controllers

{

[RoutePrefix("weather")]

public class WeatherController : ApiController

{

private static readonly List<WeatherInfo> weather = new List<WeatherInfo>

{

new WeatherInfo

{

City = "Dublin",

Conditions = "Sunny",

MaxTemperature = 22,

MinTemperature = 17,

Outlook = "Rain",

WindDirection = "North",

WindSpeed = 5

},

new WeatherInfo

{

City = "New York",

Conditions = "Snow",

MaxTemperature = 0,

MinTemperature = 0,

Outlook = "Snow",

WindDirection = "East",

WindSpeed = 20

},

new WeatherInfo

{

City = "London",

Conditions = "Rain",

MaxTemperature = 15,

MinTemperature = 5,

Outlook = "Rain",

WindDirection = "West",

WindSpeed = 40

}

};

[Route("all")]

[HttpGet]

public IHttpActionResult RetrieveAllWeatherInformation()

{

//Returning All weather information in cirty order.

return Ok(weather.OrderBy(w => w.City).ToList());

}

[Route("city/{cityName:alpha}")]

public IHttpActionResult GetWeatherInformationForCity(string cityName)

{

var cityWeather = weather.FirstOrDefault(w => w.City.ToUpper() == cityName.ToUpper());

if (cityWeather == null)

return NotFound();

return Ok(cityWeather);

}

}

}

namespace Weather.Models

{

public class WeatherInfo

{

public string City { get; set; }

public string Conditions { get; set; }

public double MaxTemperature { get; set; }

public double MinTemperature { get; set; }

public string WindDirection { get; set; }

public int WindSpeed { get; set; }

public string Outlook { get; set; }

}

}